C++ ASSIGNMENT

1.Ques :You are given the head of a linked list. Delete the middle node, and return the head of the modified linked list. [Leetcode 2095]

Ans:

```
class Solution {
public:
  ListNode* deleteMiddle(ListNode* head) {
    ListNode* slow=head;
    ListNode* fast=head;
    if(head->next==NULL) return NULL;
    while(fast!=NULL && fast->next!=NULL){
      slow=slow->next;
      fast=fast->next->next;
    }
    ListNode* prev=NULL,*curr=head;
    while(curr!=slow){
      prev=curr;
      curr=curr->next;
    }
    prev->next=curr->next;
    return head;
  }
};
```

2.Ques :You are given two linked lists: list1 and list2 of sizes n and m respectively. Remove list1 's nodes from the ath node to the bth node, and put list2 in their place. [Leetcode 1669]

Ans:

```
class Solution {
```

```
public:
  ListNode* mergeInBetween(ListNode* list1, int a, int b, ListNode*
list2) {
      ListNode* t1=list1;
      ListNode* t2=list1;
      while(a>1 && t1!=NULL){
        a--;
        t1=t1->next;
      }
      while(b>0 && t2!=NULL){
        b--;
        t2=t2->next;
      }
      t2=t2->next;
      ListNode* t3=list2;
      ListNode* t4=list2;
      while(t4->next!=NULL){
        t4=t4->next;
      }
      t1->next=t3;
      t4->next=t2;
      return list1;
   }
};
```

3.Ques :You are given the head of a linked list, and an integer k . Return the head of the linked list after swapping the values of the kth node from the beginning and the kth node from the end (the list is 1-indexed). [Leetcode 1721]

Ans:

```
class Solution {
public:
```

```
  ListNode* swapNodes(ListNode* head, int k) {
    ListNode* temp=head;
    k--;
    while(k--){
      temp=temp->next;
    }
    ListNode* p1=temp->next,*p2=head;
    while(p1){
      p1=p1->next;
      p2=p2->next;
    }
    swap(temp->val,p2->val);
    return head;
  }
};
```

4.Ques: Given the head of a linked list and an integer val , remove all the nodes of the linked list that has Node.val == val , and return the new head.

Ans:

```
class Solution {
public:
    ListNode* removeElements(ListNode* head, int val) {
    ListNode *curr = head;
    while(curr and curr->val == val){
        curr = curr->next;
    }
    head = curr;
    while(curr){
        if(curr->next and curr->next->val == val)
            curr->next = curr->next->next;
```

```
            else curr = curr->next;
        }
            return head;
        }
};
```

5.Ques:Find the length of loop in Cycle of Linked List.

Ans:

```
#include<iostream>
using namespace std;
class Node{
public:
        int val;
        Node* next;
        Node(int val){
            this->val=val;
            this->next=NULL;
        }
};
class LinkedList{
public:
        Node* head;
        int size;
        LinkedList(){
            head=NULL;
            size=0;
        }
};
void Display(Node* t){
            while(t!=NULL){
                cout<<t->val<<" ";
                t=t->next;
```

```
        }
        cout<<endl;
}
int findlength(Node* head){
    Node* fast=head->next;
    Node* slow=head;
    int fl=0;
    while(fast!=NULL && fast->next!=NULL){
        if(fast==slow){
            fl=1;
            break;
        }
        slow=slow->next;
        fast=fast->next->next;
    }
        if(fl==0) return 0;
        int cnt=1;
        slow=slow->next;
        while(slow!=fast){
            cnt++;
            slow=slow->next;
        }
        return cnt;
}
int main(){
    Node *a=new Node(1);
    Node *b=new Node(2);
    Node *c=new Node(3);
    Node *d=new Node(4);
    Node *e=new Node(5);
```

```
    a->next=b;
    b->next=c;
    c->next=d;
    d->next=e;
    e->next=c;
    Node* head=a;
    cout<<findlength(a)<<endl;
}
```